

[illegible]

```
MM      MM      000000  DDDDDDDD  IIIIII  FFFFFFFF  YY      YY
MM      MM      000000  DDDDDDDD  IIIIII  FFFFFFFF  YY      YY
MMM     MMM     00      00  DD      DD  II      FF      YY      YY
MMM     MMM     00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      00      00  DD      DD  II      FF      YY      YY
MM      MM      000000  DDDDDDDD  IIIIII  FF      YY      YY
MM      MM      000000  DDDDDDDD  IIIIII  FF      YY      YY
                                     ....
                                     ....
                                     ....
                                     ....
```

```
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```



```
0001 0 MODULE MODIFY (
0002 0
0003 0     LANGUAGE (BLISS32),
0004 0     IDENT = 'V04-001'
0005 0 ) =
0006 1 BEGIN
0007 1
0008 1 *****
0009 1 *
0010 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 *  ALL RIGHTS RESERVED.
0013 1 *
0014 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 *  TRANSFERRED.
0020 1 *
0021 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 *  CORPORATION.
0024 1 *
0025 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1 ++
0032 1
0033 1 FACILITY: F11ACP Structure Level 2
0034 1
0035 1 ABSTRACT:
0036 1
0037 1     This routine implements the MODIFY function.
0038 1
0039 1 ENVIRONMENT:
0040 1
0041 1     STARLET operating system, including privileged system services
0042 1     and internal exec routines.
0043 1
0044 1 --
0045 1
0046 1
0047 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 6-Jan-1977 23:03
0048 1
0049 1 MODIFIED BY:
0050 1
0051 1     V04-001 ACG65998 Andrew C. Goldstein, 6-Sep-1984 15:33
0052 1     Checksum file header after setting revision date
0053 1
0054 1     V03-014 CDS0007 Christian D. Saether 14-Aug-1984
0055 1     Modify handling of extension fcbs.
0056 1
0057 1     V03-013 CDS0006 Christian D. Saether 22-Apr-1984
```

```
58      0058 1      Modify access arbitration.
59      0059 1
60      0060 1      V03-012 ACG0412      Andrew C. Goldstein,      22-Mar-1984  18:26
61      0061 1      Implement agent access mode support; add access mode to
62      0062 1      protection check call
63      0063 1
64      0064 1      V03-011 CDS0005      Christian D. Saether      19-Dec-1983
65      0065 1      Use BIND_COMMON macro to reduce number
66      0066 1      of external COMMON declarations.
67      0067 1
68      0068 1      V03-010 CDS0004      Christian D. Saether      25-Sep-1983
69      0069 1      Modify SERIAL_FILE interface.
70      0070 1
71      0071 1      V03-009 CDS0003      Christian D. Saether      25-Sep-1983
72      0072 1      Manually merge ACG0343, ACG56916.
73      0073 1
74      0074 1      V03-008 ACG0343      Andrew C. Goldstein,      19-Jul-1983  16:44
75      0075 1      Inhibit revision counting if NORECORD is specified
76      0076 1
77      0077 1      V03-007 ACG56916      Andrew C. Goldstein,      21-Jun-1983  14:11
78      0078 1      Update file revision and expiration when modified
79      0079 1
80      0080 1      V03-006 LMP0149      L. Mark Pilant,      13-Sep-1983  11:27
81      0081 1      Correct a logic problem that caused problems during the
82      0082 1      protection check of a write attribute operation.
83      0083 1
84      0084 1      V03-005 CDS0002      Christian D. Saether      4-May-1983
85      0085 1      Add call to SERIAL_FILE to interlock file processing.
86      0086 1
87      0087 1      V03-004 CDS0001      Christian D. Saether      20-Apr-1983
88      0088 1      Changes to arbitrate extend and truncate access
89      0089 1      in a cluster. Truncate now requires exclusive
90      0090 1      access to the file to succeed.
91      0091 1      Change interface to TRUNCATE routine.
92      0092 1
93      0093 1      V03-003 LMP0059      L. Mark Pilant,      21-Dec-1982  11:37
94      0094 1      Always create an FCB for a file header. This eliminates
95      0095 1      a lot of special case FCB handling.
96      0096 1
97      0097 1      V03-002 LMP0036      L. Mark Pilant,      17-Aug-1982  10:15
98      0098 1      Add support for ACL's.
99      0099 1
100     0100 1      V03-001 ACG0282      Andrew C. Goldstein,      6-Apr-1982  16:08
101     0101 1      Check for device write-locked before attempting operations
102     0102 1
103     0103 1      V02-007 ACG0223      Andrew C. Goldstein,      17-Nov-1981  21:49
104     0104 1      Allow modification of directory version limit
105     0105 1
106     0106 1      V02-006 ACG0171      Andrew C. Goldstein,      7-May-1980  18:34
107     0107 1      Condition check of truncate lock count on presence of FCB
108     0108 1
109     0109 1      V02-005 ACG0167      Andrew C. Goldstein,      16-Apr-1980  19:27
110     0110 1      Previous revision history moved to F11B.REV
111     0111 1      **
112     0112 1
113     0113 1
114     0114 1      LIBRARY 'SYS$LIBRARY:LIB.L32';
```


MODIFY
V04-001

: 115
: 116

0115 1 REQUIRE 'SRC\$:FCPDEF.B32';
1106 1

B 12
16-Sep-1984 00:46:29
14-Sep-1984 12:30:37

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[F11X.SRC]MODIFY.B32;2 Page 3 (1)

MOU
V04


```
1107 1 GLOBAL ROUTINE MODIFY : L_NORM =
1108 1
1109 1 ++
1110 1
1111 1 FUNCTIONAL DESCRIPTION:
1112 1
1113 1     This routine implements the MODIFY function.
1114 1
1115 1 CALLING SEQUENCE:
1116 1     MODIFY ()
1117 1
1118 1 INPUT PARAMETERS:
1119 1     NONE
1120 1
1121 1 IMPLICIT INPUTS:
1122 1     CURRENT_WINDOW: window for file
1123 1     PRIMARY_FCB: FCB of file
1124 1     IO_PACKET: I/O packet in process
1125 1     FICE_HEADER: address of current file header
1126 1
1127 1 OUTPUT PARAMETERS:
1128 1     NONE
1129 1
1130 1 IMPLICIT OUTPUTS:
1131 1     NONE
1132 1
1133 1 ROUTINE VALUE:
1134 1     NONE
1135 1
1136 1 SIDE EFFECTS:
1137 1     file and FCB modified
1138 1
1139 1 --
1140 1
1141 2 BEGIN
1142 2
1143 2 LOCAL
1144 2     FCB_CREATED,      ! flag indicating new FCB created
1145 2     FIND_MODE,      ! mode for FIND call
1146 2     ABD              : REF BBLOCKVECTOR [ABD$C_LENGTH],
1147 2                        ! buffer descriptors
1148 2     FIB              : REF BBLOCK,      ! FIB
1149 2     FCB              : REF BBLOCK,      ! FCB of file
1150 2     HEADER           : REF BBLOCK,      ! file header
1151 2     STATUS;          ! Routine exit status
1152 2
1153 2 BIND_COMMON;
1154 2
1155 2 EXTERNAL ROUTINE
1156 2     REBLD_PRIM_FCB : L_NORM NOVALUE, ! rebuild primary fcb from header
1157 2     BUILD_EXT_FCBS : L_NORM NOVALUE, ! build extension fcb chain.
1158 2     ARBITRATE_ACCESS : [JSB_2ARGS, ! arbitrate access interlocks.
1159 2     CONV_ACCLOCK    : L_NORM,      ! convert file access lock
1160 2     LOCK_COUNT      : L_NORM,      ! determine number of locks.
1161 2     SERIAL_FILE     : L_NORM,      ! interlock file processing.
1162 2     GET_FIB         : L_NORM,      ! get FIB of request
1163 2     GET_LOC_ATTR    : L_NORM,      ! get placement data from attribute list
```



```

175 1164 2      GET LOC      : L_NORM,      | get placement data
176 1165 2      FIND       : L_NORM,      | find name in directory
177 1166 2      SWITCH_VOLUME : L_NORM,      | switch context to right volume
178 1167 2      SEARCH_FCB  : L_NORM,      | search FCB list
179 1168 2      READ_HEADER : L_NORM,      | read file header
180 1169 2      CREATE_FCB  : L_NORM,      | create an FCB
181 1170 2      CHECK_PROTECT : L_NORM,      | check file protection
182 1171 2      SET_REVISION : L_NORM,      | set file revision end expiration
183 1172 2      WRITE_ATTRIB : L_NORM,      | write file attributes
184 1173 2      EXTEND      : L_NORM,      | extend file
185 1174 2      TRUNCATE    : L_NORM,      | truncate file
186 1175 2      CHECKSUM    : L_NORM,      | checksum the file header
187 1176 2      UPDATE_FCB  : L_NORM;      | rebuild fcb from header
188 1177 2
189 1178 2
190 1179 2      ! First find the buffer descriptor, FIB, FCB, etc., and read the header.
191 1180 2      !
192 1181 2
193 1182 2      ! pointer to buffer descriptors
194 1183 2      ABD = .BBLOCK [.IO_PACKET[IRP$L_SVAPTE], AIB$L_DESCRIPTOR];
195 1184 2      FIB = GET_FIB (.ABD);
196 1185 2      IF .FIB[FIB$W_VERLIMIT] GTRU 32767
197 1186 2      THEN ERR_EXIT (SS$_BADPARAM);
198 1187 2
199 1188 2      IF .FIB[FIB$W_ALLOCATR]
200 1189 2      THEN GET_LOC_ATTR (.ABD, .FIB);
201 1190 2      GET_LOC (.FIB, LOC_RVN, LOC_LBN);
202 1191 2
203 1192 2      ! If a directory ID is present, do a directory search first.
204 1193 2      !
205 1194 2
206 1195 2      FIND_MODE = 0;
207 1196 2      IF .FIB[FIB$W_VERLIMIT] NEQ 0
208 1197 2      THEN FIND_MODE = 2;
209 1198 2      IF .CLEANUP_FLAGS[CLF_DIRECTORY]
210 1199 2      THEN FIND (.ABD, .FIB, FIND_MODE);
211 1200 2      SWITCH_VOLUME (.FIB[FIB$W_FID_RVN]);
212 1201 2
213 1202 2      ! If there is a file open on the channel, check the file ID returned by the
214 1203 2      ! FIND against the file ID that is open. If they are different, drop the FCB
215 1204 2      ! and window addresses on the floor.
216 1205 2      !
217 1206 2
218 1207 2      IF .PRIMARY_FCB NEQ 0
219 1208 2      THEN
220 1209 2          IF .PRIMARY_FCB[FCB$W_FID_NUM] NEQ .FIB[FIB$W_FID_NUM]
221 1210 2          OR .PRIMARY_FCB[FCB$W_FID_RVN] NEQ .FIB[FIB$W_FID_RVN]
222 1211 2          THEN
223 1212 2              BEGIN
224 1213 2                  PRIMARY_FCB = 0;
225 1214 2                  CURRENT_WINDOW = 0;
226 1215 2                  END;
227 1216 2
228 1217 2      ! Synchronize further processing on this FID.
229 1218 2      !
230 1219 2
231 1220 2      PRIM_LCKINDX = SERIAL_FILE (FIB [FIB$W_FID]);
```



```
232 1221 2 FCB = SEARCH_FCB (FIB[FIB$W_FID]);
233 1222 2 HEADER = READ_HEADER (FIB[FIB$W_FID], .FCB);
234 1223 2
235 1224 2
236 1225 2 ! At this point build the necessary FCB list if the file is not accessed.
237 1226 2 ! This is necessary to allow the ACL to be built.
238 1227 2
239 1228 2
240 1229 2 FCB_CREATED = 0;
241 1230 2 IF .FCB EQL 0
242 1231 2 THEN
243 1232 2 BEGIN
244 1233 2     FCB_CREATED = 1;
245 1234 2     FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
246 1235 2 END;
247 1236 2 PRIMARY_FCB = .FCB; ! Record FCB for external use
248 1237 2
249 1238 2 ! If the file is multi-header, read the extension headers and create
250 1239 2 ! extension FCB's as necessary. Finally, read back the primary header.
251 1240 2
252 1241 2
253 1242 2 IF .FCB_CREATED
254 1243 2 THEN
255 1244 2     BUILD_EXT_FCBS (.HEADER)
256 1245 2 ELSE
257 1246 2     IF .FCB [FCB$V_STALE]
258 1247 2     THEN
259 1248 2         BEGIN
260 1249 2             REBLD_PRIM_FCB (.FCB, .HEADER);
261 1250 2
262 1251 2             BUILD_EXT_FCBS (.HEADER);
263 1252 2
264 1253 2         END;
265 1254 2
266 1255 2
267 1256 2 ! Check that the volume is write enabled.
268 1257 2
269 1258 2
270 1259 2 IF .BBLOCK [CURRENT_UCB[UCB$L_DEVCHAR], DEV$V_SWL]
271 1260 2 THEN ERR_EXIT (SS$_WRITLCK);
272 1261 2
273 1262 2 ! Arbitrate access interlocks. If this is the accessor, then the file
274 1263 2 ! must be write accessed. Count a write to the file.
275 1264 2
276 1265 2
277 1266 2 IF .CURRENT_WINDOW NEQ 0
278 1267 2 THEN
279 1268 2     BEGIN
280 1269 2
281 1270 2     IF NOT .CURRENT_WINDOW[WCBS$V_WRITE]
282 1271 2     THEN ERR_EXIT (SS$_NOPRIV);
283 1272 2
284 1273 2     IF .FIB [FIB$V_TRUNC]
285 1274 2     THEN
286 1275 2         IF .FCB [FCB$W_REFCNT] NEQ 1
287 1276 2         OR LOCK_COUNT (.FCB [FCB$L_ACCLKID]) NEQ 1
288 1277 2     THEN
```



```
289      ERR_EXIT (SS$_ACCONFLICT);
290
291      IF NOT .FIB[FIB$_NORECORD]
292      THEN
293          CURRENT_WINDOW [WCBS$_WRITES] = .CURRENT_WINDOW [WCBS$_WRITES] + 1;
294
295      END
296
297      ! If it is not, then the file must not be locked against modification
298      ! and the caller must pass file protection. Count a revision to the file.
299      !
300
301      ELSE
302      BEGIN
303
304          IF .FIB [FIB$_EXTEND] OR .FIB [FIB$_TRUNC]
305          THEN
306              BEGIN
307                  LOCAL
308                  CURR_LKMODE;
309
310                  CHECK_PROTECT (WRITE_ACCESS, .HEADER, .FCB,
311                                MAXU 7, IO_PACKET[IRP$_MODE], .FIB[FIB$_AGENT_MODE]);
312
313                  CURR_LKMODE = .FCB [FCB$_ACCLKMODE];
314
315                  IF .FIB [FIB$_EXTEND]
316                  THEN
317                      BEGIN
318                          IF .FIB[FIB$_TRUNC]
319                          THEN ERR_EXIT (SS$_BADPARAM);
320
321                          IF NOT ARBITRATE_ACCESS (FIB$_WRITE, .FCB)
322                          THEN
323                              ERR_EXIT (SS$_ACCONFLICT);
324
325                          END
326                      ELSE
327
328                          ! This is a truncation. Truncation is only allowed if there is no other
329                          ! access to the file whatever.
330
331                          BEGIN
332                              IF NOT ARBITRATE_ACCESS (FIB$_NOREAD, .FCB)
333                              THEN
334                                  ERR_EXIT (SS$_ACCONFLICT);
335
336                              IF .FCB [FCB$_REFCNT] NEQ 0
337                              OR LOCK_COUNT (.FCB [FCB$_ACCLKID]) NEQ 1
338                              THEN
339                                  BEGIN
340                                      CONV_ACCLOCK (.CURR_LKMODE, .FCB);
341                                      ERR_EXIT (SS$_ACCONFLICT);
342                                  END;
343
344                          END;
345
346                  ! of trunc
347              END;
348          END;
349      END;
```

```

346 1335 4
347 1336 4      CONV_ACCLOCK (.CURR_LKMODE, .FCB);
348 1337 4
349 1338 4      END;          ! of trunc or extend
350 1339 4
351 1340 4      IF NOT .FIB[FIB$V_NORECORD]
352 1341 4      THEN
353 1342 4          BEGIN
354 1343 4              SET_REVISION (.HEADER, 1);
355 1344 4              CHECKSUM (.HEADER);
356 1345 4          END;
357 1346 4      END;
358 1347 4
359 1348 4      ! If an attribute list exists, perform the write attributes operation.
360 1349 4      !
361 1350 4
362 1351 4      IF .IO_PACKET[IRP$W_BCNT] GTR ABD$C_ATTRIB
363 1352 4      THEN
364 1353 4          BEGIN
365 1354 4              WRITE_ATTRIB (.HEADER, .ABD, 1);
366 1355 4              HEADER = .FILE_HEADER;
367 1356 4              CHECKSUM (.HEADER);
368 1357 4          END;
369 1358 4
370 1359 4      ! If the extend enable bit is on, perform the extend operation.
371 1360 4      ! If the truncate bit is on, perform the truncate operation. If both are
372 1361 4      ! on, it is an error.
373 1362 4      !
374 1363 4
375 1364 4      IF (.FIB[FIB$V_EXTEND] OR .FIB[FIB$V_TRUNC])
376 1365 4      AND .CURRENT_VCB[VCB$V_NOALLOC]
377 1366 4      THEN ERR_EXIT (SS$WRITLCK);
378 1367 4
379 1368 4      IF .FIB[FIB$V_EXTEND]
380 1369 4      THEN
381 1370 4          BEGIN
382 1371 4              EXTEND (.FIB, .HEADER);
383 1372 4          END;
384 1373 4
385 1374 4      IF .FIB[FIB$V_TRUNC]
386 1375 4      THEN
387 1376 4          TRUNCATE (.FIB, .HEADER, .FIB [FIB$L_EXVBN]);
388 1377 4
389 1378 4      HEADER = .FILE_HEADER;
390 1379 4      CHECKSUM (.HEADER);          ! checksum the file header
391 1380 4      UPDATE_FCB (.HEADER);
392 1381 4
393 1382 4      RETURN 1;
394 1383 4
395 1384 4      ! end of routine MODIFY
1 END;
```

```
.TITLE  MODIFY
.IDENT  \V04-001\
```

```
.EXTRN  REBLD PRIM_FCB, BUILD_EXT_FCBS
.EXTRN  ARBITRATE_ACCESS
```



```
.PSECT $CODE$,NOWRT,2
```

			00FC	0000G	.ENTRY	MODIFY, Save R2,R3,R4,R5,R6,R7	: 1107
	57	0000G	CF	9E	MOVAB	CHECKSUM, R7	:
	50	90	AA	D0	MOVL	-112(BASE), R0	: 1183
	56	2C	B0	D0	MOVL	@44(R0), ABD	:
			56	DD	PUSHL	ABD	: 1184
	0000G	CF	01	FB	CALLS	#1, GET_FIB	:
	52		50	D0	MOVL	R0, FIB	:
	7FFF	8F	2C	A2	CMPW	44(FIB), #32767	: 1185
			03	1B	BLEQU	1\$	:
09	16	A2	0118	31	BRW	17\$	:
			04	E1	BBC	#4, 22(FIB), 2\$	: 1188
			52	DD	PUSHL	FIB	: 1189
			56	DD	PUSHL	ABD	:
	0000G	CF	02	FB	CALLS	#2, GET_LOC_ATTR	:
			20	AA	PUSHAB	32(BASE)	: 1190
			1C	AA	PUSHAB	28(BASE)	:
			52	DD	PUSHL	FIB	:
	0000G	CF	03	FB	CALLS	#3, GET_LOC	:
			50	D4	CLRL	FIND_MODE	: 1195
			2C	A2	TSTW	44(FIB)	: 1196
			03	13	BEQL	3\$	:
	50		02	D0	MOVL	#2, FIND_MODE	: 1197
0B	6A		06	E1	BBC	#6, (BASE), 4\$	: 1198
			50	DD	PUSHL	FIND_MODE	: 1199
			52	DD	PUSHL	FIB	:
			56	DD	PUSHL	ABD	:
	0000G	CF	03	FB	CALLS	#3, FIND	:
	7E	08	A2	3C	MOVZWL	8(FIB), -(SP)	: 1200
	0000G	CF	01	FB	CALLS	#1, SWITCH VOLUME	:
	50	08	AA	D0	MOVL	8(BASE), R0	: 1207
			11	13	BEQL	6\$	:
	04	A2	24	A0	CMPW	36(R0), 4(FIB)	: 1209
			07	12	BNEQ	5\$	:
	08	A2	28	A0	CMPW	40(R0), 8(FIB)	: 1210
			03	13	BEQL	6\$	:
			08	AA	CLRQ	8(BASE)	: 1213
			04	A2	PUSHAB	4(FIB)	: 1220
	0000G	CF	01	FB	CALLS	#1, SERIAL FILE	:
	18	AA	50	D0	MOVL	R0, 24(BASE)	:
			04	A2	PUSHAB	4(FIB)	: 1222
	0000G	CF	01	FB	CALLS	#1, SEARCH_FCB	:
	53		50	D0	MOVL	R0, FCB	:
			53	DD	PUSHL	FCB	: 1223
			04	A2	PUSHAB	4(FIB)	:
	0000G	CF	02	FB	CALLS	#2, READ_HEADER	:

	54		50	D0	00099	MOVL	R0, HEADER		
			55	D4	0009C	CLRL	FCB_CREATED		1229
			53	D5	0009E	TSTL	FCB_CREATED		1230
			0D	12	000A0	BNEQ	7\$		
	55		01	D0	000A2	MOVL	#1, FCB_CREATED		1233
			54	DD	000A5	PUSHL	HEADER		1234
	0000G	CF	01	FB	000A7	CALLS	#1, CREATE_FCB		
			50	D0	000AC	MOVL	R0, FCB		
	08	AA	53	D0	000AF	7\$: MOVL	FCB, 8(BASE)		1236
		0B	55	E8	000B3	BLBS	FCB_CREATED, 8\$		1242
		0E	23	A3	E9	000B6	BLBC	35(FCB), 9\$	1246
			18	BB	000BA	PUSHR	#^M<R3,R4>		1250
	0000G	CF	02	FB	000BC	CALLS	#2, REBLD_PRIM_FCB		
			54	DD	000C1	8\$: PUSHL	HEADER		1252
	0000G	CF	01	FB	000C3	CALLS	#1, BUILD_EXT_FCBS		
03			94	AA	D0	000C8	9\$: MOVL	-108(BASE), R0	1259
	3B	A0	01	E1	000CC	BBC	#1, 59(R0), 10\$		
			00F4	31	000D1	BRW	28\$		
			50	AA	D0	000D4	10\$: MOVL	12(BASE), R0	1266
03			2D	13	000D8	BEQL	14\$		
	0B	A0	01	E0	000DA	BBS	#1, 11(R0), 11\$		1270
			24	BF	000DF	CHMU	#36		1271
				04	000E1	RET			
	13		17	A2	E9	000E2	11\$: BLBC	23(FIB), 12\$	1273
	01		18	A3	B1	000E6	CMPW	24(FCB), #1	1275
			61	12	000EA	BNEQ	19\$		
			48	A3	DD	000EC	PUSHL	72(FCB)	1276
	0000G	CF	01	FB	000EF	CALLS	#1, LOCK_COUNT		
			50	D1	000F4	CMPL	R0, #1		
			7F	12	000F7	BNEQ	22\$		
07			15	E0	000F9	12\$: BBS	#21, (FIB), 13\$		1280
	62		0C	AA	D0	000FD	MOVL	12(BASE), R0	1282
	50		28	A0	D6	00101	INCL	40(R0)	
			0091	31	00104	13\$: BRW	25\$		1266
			16	A2	95	00107	14\$: TSTB	22(FIB)	1293
				04	19	0010A	BLSS	15\$	
	76		17	A2	E9	0010C	BLBC	23(FIB), 24\$	
	50		90	AA	D0	00110	15\$: MOVL	-112(BASE), R0	1300
7E			00	EF	00114	EXTZV	#0, #2, 11(R0), -(SP)		
	0B	A0	2E	A2	91	0011A	CMPB	46(FIB), (SP)	
	02			04	1B	0011E	BLEQU	16\$	
	6E		2E	A2	9A	00120	MOVZBL	46(FIB), (SP)	
				53	DD	00124	16\$: PUSHL	FCB	1299
				54	DD	00126	PUSHL	HEADER	
				01	DD	00128	PUSHL	#1	
	0000G	CF	04	FB	0012A	CALLS	#4, CHECK_PROTECT		
			0B	A3	9A	0012F	MOVZBL	11(FCB), CURR_LKMODE	1302
			16	A2	95	00133	TSTB	22(FIB)	1304
			17	18	00136	BGEQ	20\$		
	03		17	A2	E9	00138	BLBC	23(FIB), 18\$	1308
			14	BF	0013C	17\$: CHMU	#20		1309
				04	0013E	RET			
	51		53	D0	0013F	18\$: MOVL	FCB, R1		1311
	50		0100	8F	3C	00142	MOVZWL	#256, R0	
			0000G	30	00147	BSBW	ARBITRATE_ACCESS		
				50	E8	0014A	BLBS	R0, 23\$	
	30		29	11	0014D	19\$: BRB	22\$		1313

51		53	D0	0014F	20\$:	MOVL	FCB, R1	1322	
50	0400	8F	3C	00152		MOVZWL	#1024, R0		
		0000G	30	00157		BSBW	ARBITRATE_ACCESS		
1B		50	E9	0015A		BLBC	R0, 22\$		
	18	A3	B5	0015D		TSTW	24(FCB)	1326	
		0D	12	00160		BNEQ	21\$		
	48	A3	DD	00162		PUSHL	72(FCB)	1327	
0000G	CF	01	FB	00165		CALLS	#1, LOCK_COUNT		
	01	50	D1	0016A		CMPL	R0, #1		
		0E	13	0016D		BEQL	23\$		
		53	DD	0016F	21\$:	PUSHL	FCB	1330	
		55	DD	00171		PUSHL	CURR_LKMODE		
0000G	CF	02	FB	00173		CALLS	#2, CONV_ACCLOCK		
		0800	8F	BF	00178	CHMU	#2048	1331	
			04	0017C		RET			
		53	DD	0017D	23\$:	PUSHL	FCB	1336	
		55	DD	0017F		PUSHL	CURR_LKMODE		
0000G	CF	02	FB	00181		CALLS	#2, CONV_ACCLOCK		
OE	62	15	E0	00186	24\$:	BBS	#21, (FIB), 25\$	1340	
		01	DD	0018A		PUSHL	#1	1343	
		54	DD	0018C		PUSHL	HEADER		
0000G	CF	02	FB	0018E		CALLS	#2, SET_REVISION		
		54	DD	00193		PUSHL	HEADER	1344	
	67	01	FB	00195		CALLS	#1, CHECKSUM		
	50	90	AA	D0	00198	25\$:	MOVL	-112(BASE), R0	1351
	05	32	A0	B1	0019C	CMPL	50(R0), #5		
		14	1B	001A0		BLEQU	26\$		
		01	DD	001A2		PUSHL	#1	1354	
		0050	8F	BB	001A4	PUSHR	#^M<R4, R6>		
0000G	CF	03	FB	001A8		CALLS	#3, WRITE_ATTRIB		
	54	04	AA	D0	001AD	MOVL	4(BASE), HEADER	1355	
		54	DD	001B1		PUSHL	HEADER	1356	
	67	01	FB	001B3		CALLS	#1, CHECKSUM		
		16	A2	95	001B6	26\$:	TSTB	22(FIB)	1364
			04	19	001B9	BLSS	27\$		
	OE	17	A2	E9	001BB	BLBC	23(FIB), 29\$		
	50	98	AA	D0	001BF	27\$:	MOVL	-104(BASE), R0	1365
05	OB	A0	04	E1	001C3	BBC	#4, 11(R0), 29\$		
		025C	8F	BF	001C8	28\$:	CHMU	#604	1366
			04	001CC		RET			
		16	A2	95	001CD	29\$:	TSTB	22(FIB)	1368
			07	18	001D0	BGEQ	30\$		
			14	BB	001D2	PUSHR	#^M<R2, R4>	1371	
0000G	CF	02	FB	001D4		CALLS	#2, EXTEND		
	OA	17	A2	E9	001D9	30\$:	BLBC	23(FIB), 31\$	1374
		1C	A2	DD	001DD	PUSHL	28(FIB)	1376	
			14	BB	001E0	PUSHR	#^M<R2, R4>		
0000G	CF	03	FB	001E2		CALLS	#3, TRUNCATE		
	54	04	AA	D0	001E7	31\$:	MOVL	4(BASE), HEADER	1378
		54	DD	001EB		PUSHL	HEADER	1379	
	67	01	FB	001ED		CALLS	#1, CHECKSUM		
		54	DD	001F0		PUSHL	HEADER	1380	
0000G	CF	01	FB	001F2		CALLS	#1, UPDATE_FCB		
	50	01	D0	001F7		MOVL	#1, R0	1382	
		04	001FA		RET			1384	

; Routine Size: 507 bytes, Routine Base: \$CODE\$ + 0000

MODIFY
V04-001

K 12
16-Sep-1984 00:46:29
14-Sep-1984 12:30:37

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[F11X.SRC]MODIFY.B32;2 Page 12
(2)

```
: 396      1385  1
: 397      1386  1 END
: 398      1387  0 ELUDOM
```

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	507	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	50	0	1000	00:02.0

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:MODIFY/OBJ=OBJ\$:MODIFY MSRC\$:MODIFY/UPDATE=(ENH\$:MODIFY)

```
: Size:      507 code + 0 data bytes
: Run Time:   00:24.2
: Elapsed Time: 00:56.4
: Lines/CPU Min: 3441
: Lexemes/CPU-Min: 39769
: Memory Used: 319 pages
: Compilation Complete
```


0171 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

